Enabling Tiled Displays for Education

Luc Renambot, Tom van der Schaaf
{renambot,tom}@cs.vu.nl
Faculty of Sciences - Vrije Universiteit
De Boelelaan 1081, 1081 HV Amsterdam, The Netherlands

1. Introduction

This document describes the interaction with the *ICWall* tiled display built at the *Vrije Universiteit*, within a project called MultiVLA. The display has been setup within an educational environment and is used for several lectures from computer science (graphics, parallel programming, AI), physics, and chemistry. We describe a system consisting of several free software components that allows novice users to display their multimedia onto the tiled display. The user interface is simple and requires no knowledge about the underlying architecture, hiding the complex parallel rendering system driving the display. Since the wall is also used for research, the system allows for flexible switching between desktop applications and virtual reality applications.

The key characteristics to choose a tiled display in a education environment are among:

- Scalability: The size of the display is determined by the number of projectors used. A small setup (around 8 projectors) can easily be used for medium-sized groups ($\approx$ 50 persons).
- Standard components: The setup can be realized using commodity (off-the-shelf) components. This reduces the price substantially.
- Resolution: The resolution of the screen is the sum of the resolution of the individual projectors.
- Size: The large format allows a wide variety of different applications. An advantage is the saturation of field of view and the possibility for simultaneous projection of different types of information.

All these characteristics make a video wall in principal more apt to be used in educational environments than for instance a CAVE or a standard projector. Within the academia, there is a growing interest for the development and use of scalable displays as a complementary technique to CAVE systems. The main issues are both the technical and didactic aspects of hardware and software developed. Interaction and collaboration greatly determine the success of the project: interaction between the lecturer and the display, interaction with the students or the audience, local collaboration between the lecture room and our AccessGrid [1] node, and collaboration with remote sites.

Figure 1. Overview of the classroom

The global view for the room of the tiled display is given in Figure 1. The approach is that the room should be as multi-functional as possible. In the current design, several components are present, such as the large projection screen, a plasma panel with touch screen (Smartboard), and wireless network. The lecturer uses the touch screen as interface for control and interaction of the tiled display. The room is capable of seating about 50 people for a projection area of roughly 5.0 by 2.5 meters. Sliding walls make a reorganization of the room possible. Also, we plan to use the room as an AccessGrid node (dedicated applications are under development).

2. Interaction and User Interface

Our display will be used extensively for teaching purposes, bridging the gap between the laboratory and the

classroom. Researchers are primarily interested in using tiled displays for visualization of complex simulations or high-resolution data. Teachers are more interested in showing their presentations, movies, and websites. In order to fulfill these different needs, the setup must meet several requirements:

The system

- The hardware must appear as a single host with a single (albeit very large) monitor. This implies that the parallelism of the underlying graphics cluster needs to be transparent.
- The system should support as many applications as possible, from presentation and desktop tools to interactive 3D applications. Possibility to visualize standard file formats is an important demand from teachers (for instance, VRML for 3D scene or PDB molecule for protein visualization). Performance should be at least comparable to a standard environment (single PC), and hopefully better using the resources (computing and graphics) available on the driving cluster.

The user interface

- It should be possible to start programs with a single mouse click (for instance, using a web interface). Users should not require high-level computer skills to run their programs.
- The human-computer interface should be open to accept a large variety of input to accommodate advanced users or demanding applications (for instance, joystick, PDA, tracking devices, speech system).

For the implementation of the above described environment, we made use of the following free software packages:

- **Parallel Aura** for the low level graphics driving the cluster,
- **Apache web server** to host the user interface (java applet) and documentation,
- **VNC** offering a remote display system for displaying a large desktop. We use the following packages to support popular media:
  - **OpenOffice** for displaying presentations (for instance, MS powerpoint),
  - **Acrobat Reader** for displaying PDF documents,
  - **Mozilla** for web browsing,
  - **Mplayer** for showing movies in numerous file formats.
- **XML-RPC** (in Java, C++ and Python) to serve as software glue between all the different distributed components.

Hiding the parallelism, is solved by using our 3D graphics API called Aura [2, 4]. Aura transparently solves many problems induced by a parallel graphics cluster, i.e. distributing graphics, geometrical alignment of the projectors, as well as color and intensity adjustment of the projectors. To provide a standard desktop within Aura, we use a virtual display server called VNC [3]. A VNC server receives requests from clients, upon which it returns screen updates of the virtual desktop. Changes to the screen are compressed and sent to the requesting client. Aura has a VNC client module built in, that is optimized for parallel display. It unpacks the incoming screen updates and places them into a texture. The texture is displayed over the entire screen. Since VNC supports both Windows and Linux, we are now able to display any Windows or Linux application. An important advantage of VNC is that it is not limited to any physical constraints, and thus the desktop can be of any size or bit depth. The downside of this approach is that demanding applications, such as movies, can be slow.

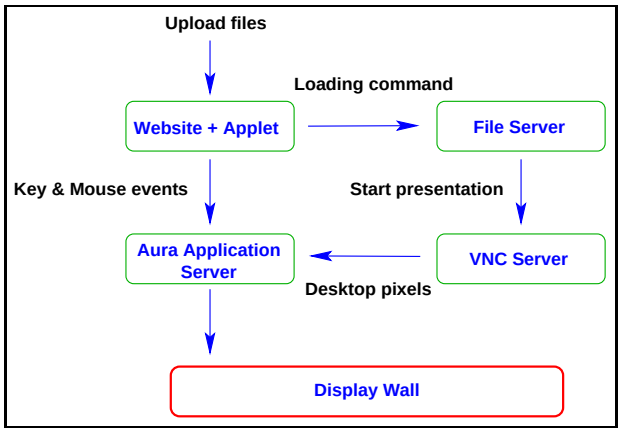

Figure 2. Interface and Interaction System

Apart from the technical perspective, there is the user perspective. Not all users are experienced with Linux command line, so a simple user interface should be provided. Therefore, our wall can be fully controlled from a website, as shown in Figure 2. Teachers can upload their presentations, movies, and datasets to their account via the website. After the upload, their new file is immediately available. Simply clicking on the file displays it on the wall. To achieve this behavior, we use a combination of two servers and a java client (Figure 3 shows a picture of the java applet). To link these different components, all communication is made through standard XML-RPC.

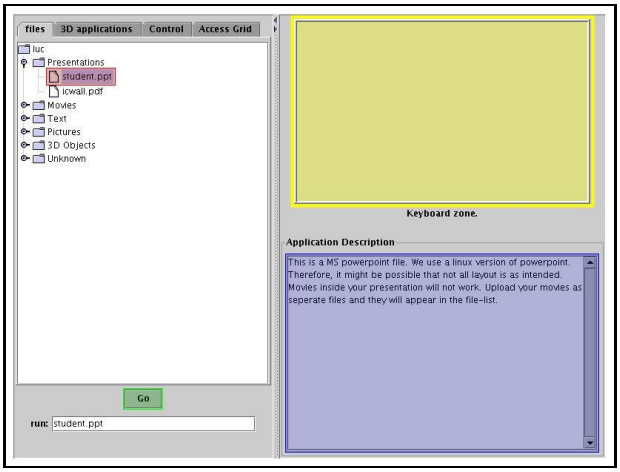

Figure 3. The java ICWall control applet

The first server is a file manager (written in Python). It handles all file issues: uploading, starting programs in the VNC desktop, and giving account info to the web client.

The second server is the Aura application server that will show the output of the program. This server, upon request, loads the dynamically loadable library matching the request (e.g. a VNC desktop if the request is a Powerpoint presentation, or the Aura image viewer if the request is an image). The loaded library uses the parallel Aura API to provide the requested functionality. Upon loading a new library, the previous one is discarded. However, we plan to support multiple libraries running concurrently for a better use of the projection surface. Currently, we implemented several libraries:

- An image viewer to display images over the whole screen (jpg, gif, ...).
- An object viewer to display common 3D file types (obj, nff, lwo, ...).
- A Linux VNC client to display all other media types. We use OpenOffice, Mplayer, Mozilla and Acrobat Reader to support most popular media (ppt, pdf, mpeg, avi, ...). Figure 4 shows a lecturer using a desktop with a powerpoint presentation and a movie.
- Several virtual reality applications, each implemented as a separate library. Figure 5 shows the same lecturer, demonstrating an interactive visualization of a tooth.

In the future, we will write more libraries. For example, movies can be fairly slow using the VNC server (especially high-resolution movies). We will write a separate movie player for Aura that uses low-level features to improve performance. Movie files can then be redirected to the new library instead of the VNC desktop.

Figure 4. An actual lecture

Figure 5. A lecturer running a 3D visualization

The user can control the application using through the website. Mouse moves and keyboard strokes are transmitted to the application server and passed to the library to interpret them. Experienced Linux users can use a local VNC client (via the website), that allows full control, so that they can start and control any Linux program themselves, bypassing the website.

3. Conclusion

We built a user friendly interface to a tiled display wall, that hides all technical details. We do not have enough day-to-day experience with users to make a full usability report yet, but the first lecturers to use it were very positive. Starting in September, the tiled display will be used on a daily basis by lecturers from different departments of the University. This should give us the required feedback to improve the system.

References

[1] L. Childers, T. Disz, B. Olson, and R. Stevens. Scalable high-resolution wide area collaboration over the Access Grid. In ACM, editor, *SC2000: High Performance Networking and Computing*, pages 121–121, Nov. 2000.

[2] D. Germans, H. J. Spoelder, L. Renambot, and H. E. Bal. VIRPI: A High-Level Toolkit for Interactive Scientific Visualization in Virtual Reality. In *5th Immersive Projection Technology Workshop (IPT98)*, May 2001.

[3] T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper. Virtual network computing. *IEEE Internet Computing*, 2(1):33–38, Jan. 1998.

[4] T. van der Schaaf, L. Renambot, D. Germans, H. Spoelder, and H. Bal. Retained Mode Parallel Rendering for Scalable Tiled Displays. In *Proc. 6th annual Immersive Projection Technology (IPT) Symposium, Orlando Florida*, Mar. 2002.